REPAST MODEL TESTING GUIDE

JONATHAN OZIK, NICK COLLIER - REPAST DEVELOPMENT TEAM

0. Before we Get Started

Before we can do anything with Repast Simphony, we need to make sure that we have a proper installation of Repast Simphony 2.1. Instructions on downloading and installing Repast Simphony on various platforms can be found on the Repast website.

1. Getting Started with Repast Simphony Model Testing

This guide will walk you through a few model testing use cases using Repast Simphony. To learn more about model testing, including the benefits of test driven development (TDD) when developing agent models, see Collier and Ozik (2013)[1].

To add tests into an existing Repast Simphony project, we recommend the following setup steps:

(1) Add a `test` source folder to the project. This can be accomplished in a number of ways. One way is to right click on the project and navigate to *New* → *Source Folder* (Fig. 1). Then fill in `test` in the *Folder name* text field and click on *Finish* (Fig. 2).

(2) Modify the output folder for the `test` source folder. Right click on the project and navigate to *Properties*. Then choose the *Java Build Path* entry in the left bar and select the *Source* tab. Select the check box that reads *Allow output folders for source folders* and expand the `test` entry (Fig. 3). Select the *Output folder* entry and click on the *Edit* button (Fig. 4). Then select the *Specific output folder* option and input `testbin`, or something else different from `bin` (Fig. 5). Click on *Okay* and then *Okay* again.

(3) Add a JUnit Test Case by right clicking on the project in the Package Explorer view, choosing *New* → *Other*... (Fig. 6). Then navigate to *Java* → *JUnit* and choose *Junit Test Case* (Fig. 7). In the *New JUnit Test Case* wizard choose the *New JUnit 4 test* option, specify the correct source folder (`test`), name your test case, include all the method stubs, and click on *Finish* (Fig. 8). If JUnit was not previously on your project's build path, you will see a dialog asking if you'd like to add it (Fig. 9). Click *OK* to add the JUnit 4 library to the project's build path.

Date: June 20, 2013.

[1]Collier, N, and J Ozik. Test-Driven Agent-Based Simulation Development. To appear in WSC 2013 Proceedings. Washington, D.C., 2013.

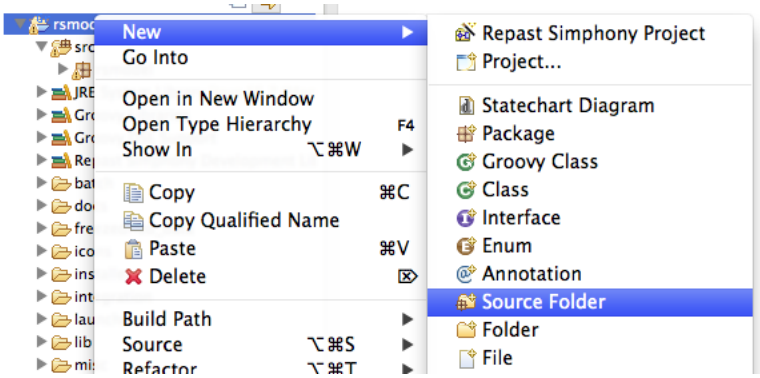

FIGURE 1. Selecting your project, right-clicking and choosing *New* → *Source Folder*.

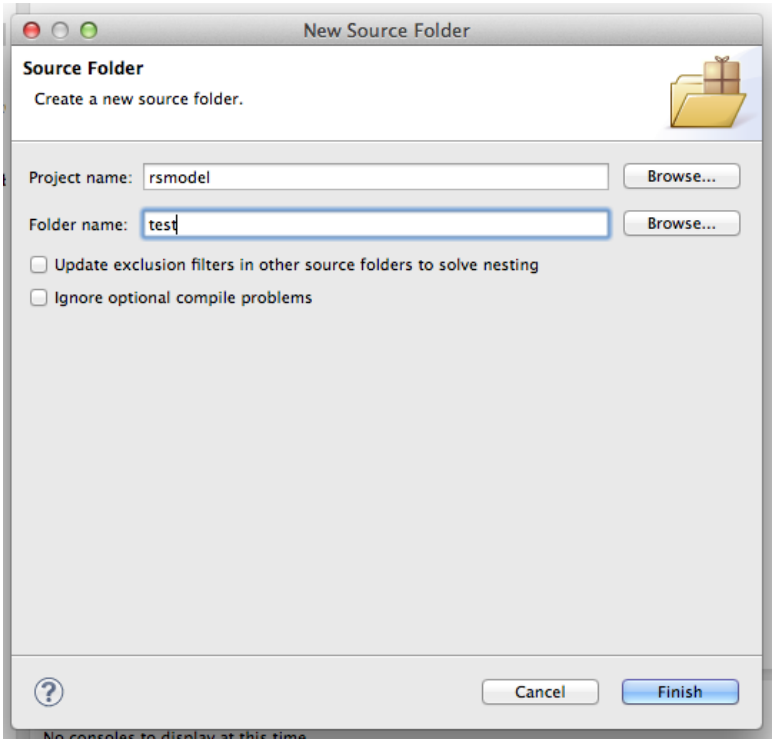

FIGURE 2. New source folder wizard.

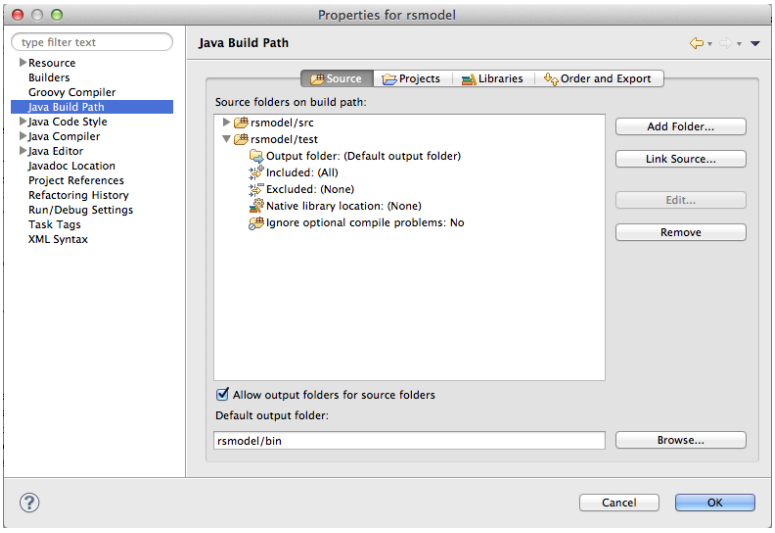

FIGURE 3. *Java Build Path* → *Source* tab in a project's properties.

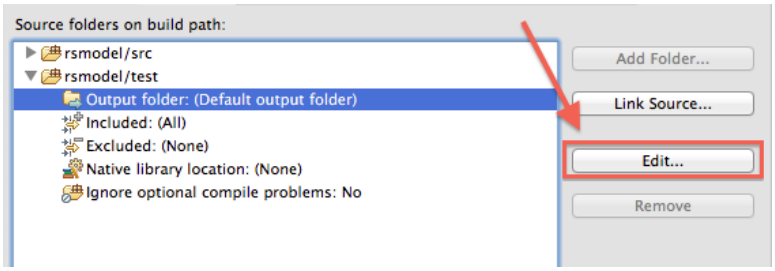

FIGURE 4. Edit the output folder of the `test` source folder.

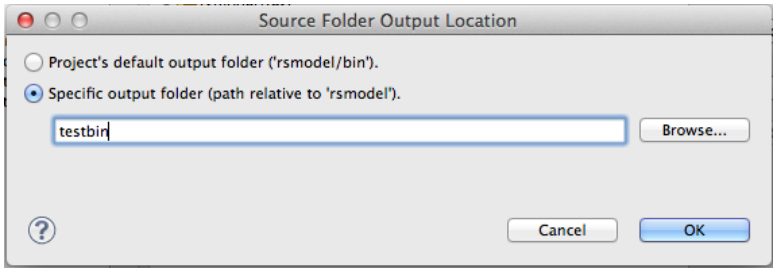

FIGURE 5. Choosing the source folder output location.

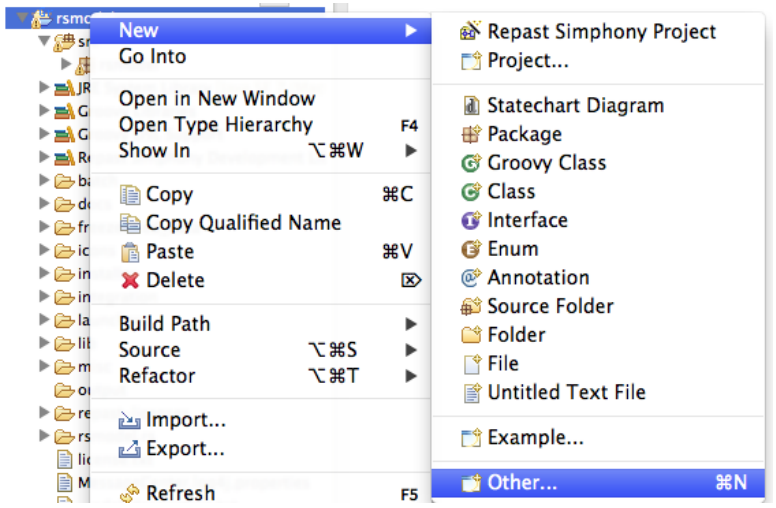

FIGURE 6. Selecting your project, right-clicking and choosing New → Other... .

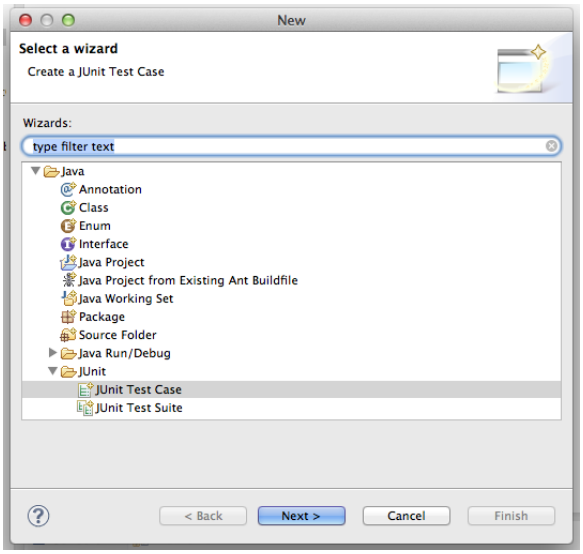

FIGURE 7. The JUnit Test Case option within Java → JUnit.

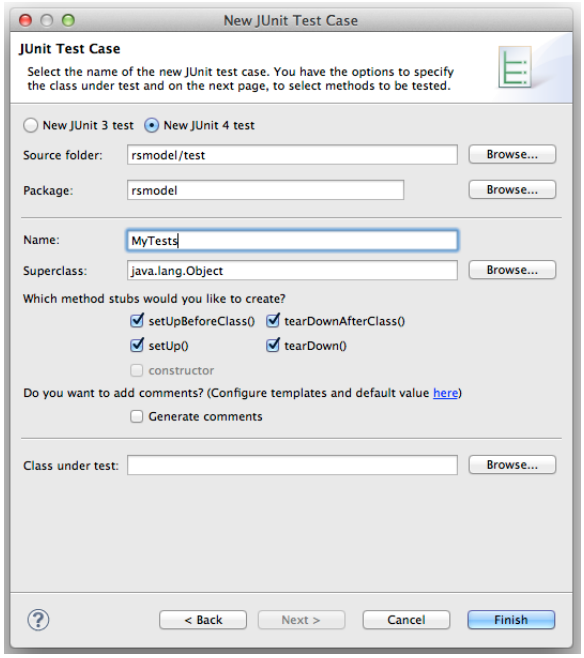

FIGURE 8. The new *JUnit Test Case* wizard.

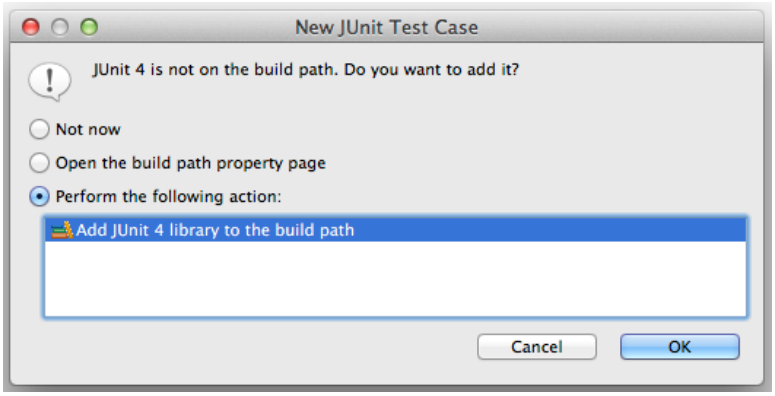

FIGURE 9. Add JUnit 4 library to build path.

Once the setup is complete, there are a number of different types of model tests that can be created, based on the nature of the model behavior that is being tested. We will go over a few such cases next.

1.1. **Use Case 1: Simple Unit Testing with Repast Simphony Models.** If the elements being tested are relatively decoupled, there is nothing special that needs to be done in terms of test case setup. In this scenario, the @BeforeClass, @AfterClass, @Before, and @After annotated methods do not need any Repast Simphony specific elements and tests can be written in the usual way JUnit tests are written.

1.2. **Use Case 2: Schedule Based Model Testing with Repast Simphony Models.**

```
1 @Before
2 public void setUp() throws Exception {
3    Schedule schedule = new Schedule();
4    RunEnvironment.init(schedule, null, null, true);
5    Context context = new DefaultContext();
6    RunState.init().setMasterContext(context);
7
8    // Any additional setup
9 }
```

LISTING 1. @Setup method in a schedule dependent test case.

1.3. **Use Case 3: Context Builder Based Model Testing with Repast Simphony Models.** For cases

```
. . .
public Context context;

@Before
public void setUp() throws Exception {
   Schedule schedule = new Schedule();
   RunEnvironment.init(schedule, null, null, true);
   context = new DefaultContext();
   MyContextBuilder builder = new MyContextBuilder();
   context = builder.build(context);
   RunState.init().setMasterContext(context);

   // Any additional setup
}
. . .
```

LISTING 2. @Setup method in a schedule dependent test case.

1.4. **Use Case 4: Model Testing with ReLogo Models.** For cases

```
1  . . .
2  static UserObserver observer;
3
4  @BeforeClass
5  public static void setUpBeforeClass() throws Exception {
6     String scenarioDirString = "ModelName.rs"
7     File paramsFile = new File(new File(scenarioDirString),
8               "parameters.xml");
9     ParametersParser pp = new ParametersParser(paramsFile);
10    Parameters params = pp.getParameters();
11    RunEnvironment.init(new Schedule(), null, params, true);
12    Context context = new DefaultContext();
13    SimBuilder builder = new SimBuilder();
14    context = builder.build(context);
15    observer = (UserObserver) context.iterator().next();
16
17    // Any additional before class setup
18 }
19
20 @Before
21 public void setUp() throws Exception {
22    observer.clearAll();
23
24    // Any additional setup
25 }
26 . . .
```

LISTING 3. @Setup method in a schedule dependent test case.